Pattern Recognition 44 (2011) 2656–2668

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

A probabilistic model of classifier competence for dynamic ensemble selection

Tomasz Woloszynski *, Marek Kurzynski

Chair of Systems and Computer Networks, Faculty of Electronics, Wroclaw University of Technology, Wybrzeze Wyspianskiego 27, 50-370 Wroclaw, Poland

ARTICLE INFO

Article history:
Received 26 August 2010
Received in revised form
16 March 2011
Accepted 17 March 2011
Available online 29 March 2011

Keywords:
Probabilistic modelling
Classifier competence
Multiple classifier system
Beta distribution

ABSTRACT

The concept of a classifier competence is fundamental to multiple classifier systems (MCSs). In this study, a method for calculating the classifier competence is developed using a probabilistic model. In the method, first a randomised reference classifier (RRC) whose class supports are realisations of the random variables with beta probability distributions is constructed. The parameters of the distributions are chosen in such a way that, for each feature vector in a validation set, the expected values of the class supports produced by the RRC and the class supports produced by a modelled classifier are equal. This allows for using the probability of correct classification of the RRC as the competence of the modelled classifier. The competences calculated for a validation set are then generalised to an entire feature space by constructing a competence function based on a potential function model or regression. Three systems based on a dynamic classifier selection and a dynamic ensemble selection (DES) were constructed using the method developed. The DES based system had statistically significant higher average rank than the ones of eight benchmark MCSs for 22 data sets and a heterogeneous ensemble. The results obtained indicate that the full vector of class supports should be used for evaluating the classifier competence as this potentially improves performance of MCSs.

1. Introduction

Multiple classifier systems (MCSs) were shown to outperform single classifiers for a wide range of classification problems [1–4]. The reason is that a combination of classifiers reduces risks associated with picking an inadequate single classifier, choosing a space of classifiers not containing the optimal classifier, and falling into local error minima during training [5–7]. However, in order for an ensemble of classifiers to perform better than an individual classifier, the ensemble has to be diverse (i.e. the classifiers have to make independent errors) and the combination method used has to effectively exploit that diversity [8,9]. One possible way to achieve diversity of the ensemble is to generate different training sets for the classifiers through, for examples, bootstrapping [10], boosting [11] and random subspaces [12]. Another way is to use a heterogeneous ensemble of structurally diverse classifiers [4,13,14].

For a combination of classifiers, two approaches used are classifier fusion (CF) and classifier selection (CS). In the CF approach, a test object is classified using a combination function and all classifiers in the ensemble. The combination functions used are sum, product, maximum, minimum, majority voting, fuzzy integral, and others [6,8,9]. However, redundant and inaccurate classifiers in the ensemble can adversely affect performance of a system based on the combination functions. This is because redundant classifiers reduce diversity of the ensemble and subsequently they only increase complexity of the system [13]. Also, performance of the system is unlikely to improve if inaccurate classifiers are included in the combination process. To remedy this, ensemble pruning (EP) methods have been developed [15–18]. The methods are based on selecting and combining a subset of classifiers from the ensemble instead of combining all. The selection criteria used are diversity [6,19] and performance [20] of the selected subset, and a mixture of the two [13,21]. For small ensembles, the optimal subset can be found through exhaustive search. For large ensembles, a quasi-optimal subset is found using heuristic and hill-climbing optimisation algorithms, e.g. genetic algorithms [13,15], reinforcement learning [22] and quadratic integer programming [23]. However, the subset selection in the EP methods is independent on the location of the test object in a feature space. Consequently, there may exist a different subset that locally performs better than the subset selected globally.

In the CS approach, the test object is classified by a single classifier that is statically or dynamically selected from the

* Corresponding author.
E-mail addresses: tomasz.woloszynski@pwr.wroc.pl (T. Woloszynski), marek.kurzynski@pwr.wroc.pl (M. Kurzynski).

0031-3203/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2011.03.020

ensemble. In the static classifier selection, first each classifier in the ensemble is assigned with a region of competence in the feature space during training. Then, the classifier assigned with the region of competence that contains the test object is selected [24,25]. In the dynamic classifier selection (DCS), first a competence of each classifier is evaluated for the test object and then the most competent classifier is selected [6,26,27]. For the calculation of the competence, various performance estimates are used [26–29]. One drawback of the CS methods is that their performance depends solely on an accurate estimation. Another drawback is that the region of competence may be more difficult to estimate than the optimal decision boundary for some simple classification problems.

Recently, dynamic ensemble selection (DES) methods that use a mixture of the CF and CS approaches have been introduced [30,31]. The methods select and combine a subset of classifiers from the ensemble for each test object. The selection criteria used are performance of individual classifiers [30] and subsets of classifiers [31]. There are also DES methods that do not require the ensembles of trained classifiers, e.g. mixtures of experts (MoE) [32]. In the MoE, classifiers are trained and their competences are calculated in a coupled manner. However, many of the DCS and DES methods are ad hoc or heuristic. Consequently, it is difficult to draw sound conclusions about possible improvements for either a specific method or MCSs in general. For this reason, there is a growing interest in the theoretical explanation and justification of approaches, methods and concepts used for classifier combination [6,8,33].

In this paper, the classifier competence is studied using a probabilistic model. The study is the continuation of the previous work on competence measures for DES based systems [34–36]. In the previous work, a support given by a classifier for the correct class was modelled by a random variable and a competence measure based on the modelling was developed. However, not all values of the support could be modelled and the measure could not be used to evaluate worse-than-random classifiers. In this study, a unified modelling of the full vector of class supports is derived. Using the modelling, a competence measure is developed that can be used to evaluate any classifier in the ensemble. Three DCS and DES based systems were constructed using the measure developed. Performance of the systems was compared against two classical combination methods (single best and majority voting) and six DCS and DES based systems such as DCS-potential function estimate (DCS-PFE) [26], DCS-local accuracy (DCS-LA) [27], DCS-modified local accuracy (DCS-MLA) [28], DCS-multiple classifier behaviour (DCS-MCB) [29], DES-K nearest oracles eliminate (DES-KE) [30], and mixtures of experts (MoE) [32]. For the comparison, 22 benchmark data sets from the UCI Machine Learning Repository [37], the Ludmila Kuncheva Collection [38] and the ELENA project [39] were used.

This paper is organised as follows. In Section 2, a probabilistic model of the classifier competence is developed. Section 3 describes the systems that were constructed using the model. The experiments conducted are shown in Section 4 and the results with discussion are presented in Section 5. The paper is concluded in Section 6.

2. Theoretical framework

2.1. Classifier ensemble

Let a set of trained classifiers $\Psi=\{\psi_1,\ldots,\psi_L\}$, called a classifier ensemble, be given and let a classifier ψ_l, $l=1,\ldots,L$ be a function $\psi_l:\mathcal{X}\to\mathcal{M}$ from a feature space $\mathcal{X}\subseteq\mathbb{R}^n$ to a set of class labels $\mathcal{M}=\{1,\ldots,M\}$. A canonical model of classification is assumed [6,40], where the classifier ψ_l produces a vector of class supports $[d_{l1}(x),\ldots,d_{lM}(x)]$ for a feature vector $x\in\mathcal{X}$. It is further assumed, without loss of generality, that $\sum_{j=1}^{M} d_{lj}(x)=1$ and $d_{lj}(x)\geq 0$. Classification is made according to the maximum rule

$$\psi_l(x)=i \;\Leftrightarrow\; d_{li}(x)=\max_{j\in\mathcal{M}} d_{lj}(x). \tag{1}$$

The ensemble Ψ is used for classification through a combination function which, for example, can select a single classifier or a subset of classifiers from the ensemble, it can be independent or dependent on the feature vector x (in the latter case the function is said to be dynamic), and it can be non-trainable or trainable. For the dynamic combination functions, the concept of a classifier competence is frequently used. A competence function $c(\psi_l,x)$ estimates performance of the classifier ψ_l for x and it usually takes values in the interval [0,1], where the value of 0 (1) indicates the least (the most) competent classifier. Ideally, the function should be easy to calculate for arbitrary numbers of classes, features, and classifiers and it should be independent on the combination function and the methods used for constructing classifiers in the ensemble. In this study, a trainable competence function with the above properties is developed using a probabilistic model. For the training of the competence function, it is assumed that a validation set $V=\{(x_1,j_1),\ldots,(x_N,j_N)\}$ containing pairs of feature vectors and their corresponding class labels is available. For the existing DCS and DES based systems, the function developed would replace the module that calculates the classifier competences using the validation set.

2.2. Measuring the classifier competence

A natural competence measure of the classifier ψ_l for the feature vector x is the probability of correct classification $P_c(\psi_l|x)$. The probability can be written as

$$P_c(\psi_l|x)=\sum_{j=1}^{M}\Pr\{x \text{ belongs to the } j\text{-th class } \wedge\ \psi_l(x)=j\}, \tag{2}$$

where $\Pr\{S\}$ is the probability that a statement S is true. However, the probability is equal to 0 or 1 unless at least one of the two terms inside the probability operator in (2) is a random event. This is true in one of the two following cases[1]:

1. A probabilistic model of classification is used, where feature vectors and class labels are realisations of a random variable pair (X,J). Using the model, the probability (2) becomes
$$P_c(\psi_l|x)=\Pr\{x \text{ belongs to the } j\text{-th class}\},\quad \text{where } \psi_l(x)=j. \tag{3}$$

2. The classifier ψ_l assigns the class label j to the feature vector x in a stochastic manner. In this case, the probability (2) becomes
$$P_c(\psi_l|x)=\Pr\{\psi_l(x)=j\},\quad \text{where } j \text{ is the class label of } x. \tag{4}$$

There are problems, however, in both cases. First, the probabilistic model of classification is often used to construct some of the classifiers in the ensemble (as it is in this study) and therefore, it should not also be used to construct the competence function. This is because no one should be a judge in their own cause[2], meaning that the use of the same learning paradigm to construct a classifier and to evaluate its competence is unfair to the

[1] The case where both terms are random events is not considered.

[2] *Nemo iudex in causa sua*, a fundamental principle of natural justice ensuring fairness of judgement.

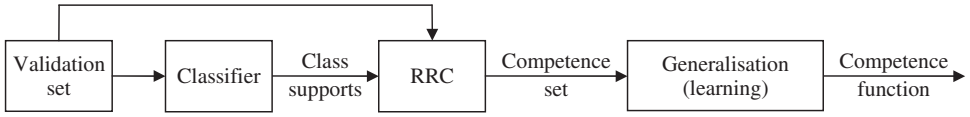

Fig. 1. Flowchart of the proposed method for calculating the competence function.

classifiers constructed using different learning paradigms. Second, the assumption that the classifier assigns a class label in a stochastic manner has little or no practical use and hence it should be avoided. For the above reasons, a direct application of the probabilistic model to the problem of calculating the classifier competence is not used in this study. Instead, an indirect method for solving the problem is developed. The method developed consists of the following two steps.

In the first step, a hypothetical classifier called a randomised reference classifier (RRC) is constructed. The classifier produces a randomised vector of class supports in such a way that its expected value is equal to the vector of class supports produced by the classifier ψ_l for each of the feature vectors x_k, $k=1,\ldots,N$ in the validation set. Consequently, the RRC can be considered as equivalent to the classifier ψ_l and its probability of correct classification $P_c(RRC|x_k)$ can be used as the competence $C(\psi_l,x_k)$ of that classifier. This observation is used to construct a competence set $\mathcal{C}_l$ for each classifier ψ_l in the ensemble

$$\mathcal{C}_l=\{(x_1,C(\psi_l,x_1)),\ldots,(x_N,C(\psi_l,x_N))\}. \tag{5}$$

In the second step, the competence set $\mathcal{C}_l$ is used to construct the competence function $c(\psi_l,x)$. The construction is based on extending (generalising) the competences $C(\psi_l,x_k)$, $k=1,\ldots,N$ to the entire feature space $\mathcal{X}$. Therefore, the construction of the competence function can be considered as a problem of learning the function using the competence set. The flowchart of the proposed method for calculating the competence function is shown in Fig. 1. The next two subsections describe the steps of the method in detail.

2.3. Randomised reference classifier (RRC)

The RRC is a stochastic classifier and therefore it is defined using a probability distribution over the set of class labels $\mathcal{M}$ [41] or, assuming the canonical model of classification, over the product of class supports $[0,1]^M$. In other words, the RRC uses the maximum rule and the vector of class supports $[\delta_1(x),\ldots,\delta_M(x)]$ for the classification of the feature vector x, where the j-th support is a realisation of a random variable (rv) $\Delta_j(x)$. The probability distributions of the rvs are chosen in such a way that the following conditions are satisfied[3]:

1. $\Delta_j(x)\in[0,1]$,

2. $\mathbf{E}[\Delta_j(x)]=d_j(x), \quad j=1,\ldots,M$,

3. $\sum_{j=1}^{M}\Delta_j(x)=1$, (6)

where $\mathbf{E}$ is the expected value operator. Conditions 1 and 3 follow from the normalisation properties of class supports while condition 2 relates the RRC to ψ, ensuring their equivalence. Since, for each x, the RRC performs classification in a stochastic manner, it is possible to calculate the conditional probability of correct classification $P_c(RRC|x)$ [41]. Assuming that x belongs to the i-th class, the probability is given by

$$P_c(RRC|x)=\Pr\left[\bigwedge_{j=1,\ldots,M,\ j\neq i}\Delta_i(x)>\Delta_j(x)\right]. \tag{7}$$

It is important to notice that the probability $P_c(RRC|x)$ is independent on the description of a classification problem at hand and it can be calculated even if the problem is formulated using a non-probabilistic model.

From the above definition, it follows that the RRC can be considered as equivalent to the classifier ψ for the feature vector x since it produces, on average, the same vector of class supports as the modelled classifier. Consequently, it is justified to use the probability $P_c(RRC|x)$ as the competence of the classifier ψ for x, i.e.

$$C(\psi,x)=P_c(RRC|x). \tag{8}$$

For the calculation of the classifier competence, the feature vector x and its class label j must be known and therefore the competences can be calculated only for the pairs (x_k,j_k), $k=1,\ldots,N$ taken from the validation set.

The key element in the modelling presented above is the choice of the probability distributions for the rvs $\Delta_j(x_k)$, $j=1,\ldots,M$ for each x_k so that conditions 1–3 in (6) are satisfied. It needs to be emphasised that, apart from the extreme case where a vector of class supports to be modelled has zeros in all positions but one (e.g. $[1,0,\ldots,0]$), the choice of the distributions is not unique. Consequently, the values of the probability $P_c(RRC|x_k)$ and the classifier competence $C(\psi,x_k)$ depend on the definition of the distributions. In this study, beta probability distributions with the parameters $\alpha_j(x_k)$ and $\beta_j(x_k)$, $j=1,\ldots,M$ are used. The choice of the beta distributions follows from the analysis of the class supports produced by a random classifier. In particular, it is shown that if the class supports of the random classifier are defined using a random division of the unit interval, then the supports must be beta distributed. This result is then used to develop a method for modelling class supports produced by any classifier. The next paragraph justifies the use of the beta distributions and the method developed in detail.

Let $\mathcal{U}$ be a set of $M-1$ uniformly distributed numbers on the interval $[0,1]$ and let z_m, $m=1,\ldots,M-1$ be a sequence of numbers taken from $\mathcal{U}$ arranged in an increasing order. Since the numbers z_m divide the interval $[0,1]$ into M subintervals, the differences $\delta_j=z_j-z_{j-1}$, $j=1,\ldots,M$, where $z_0=0$ and $z_M=1$ can be considered as class supports. From order statistics, it follows that each support δ_j is a realisation of the rv Δ_j with the beta probability distribution [42]

$$\Delta_j\sim b(u,\alpha_j,\beta_j)=\frac{u^{\alpha_j-1}(1-u)^{\beta_j-1}}{\int_0^1 u^{\alpha_j-1}(1-u)^{\beta_j-1}\,du}, \tag{9}$$

where $u\in[0,1]$, $\alpha_j=1$ and $\beta_j=M-1$. The expected value of the rv Δ_j is

$$\mathbf{E}[\Delta_j]=\frac{\alpha_j}{\alpha_j+\beta_j}=\frac{1}{M}. \tag{10}$$

It can be noticed that the parameters α_j and β_j sum to M and that the parameters and the expected values $\mathbf{E}[\Delta_j]$ are independent of j. Consequently, the rvs Δ_j, $j=1,\ldots,M$ can only model, in terms of their expected values, a vector of class supports equal to

[3] Throughout this section, the index l of the classifier ψ_l and the class supports $d_{lj}(x)$ is dropped for clarity.

$[1/M,\ldots,1/M]$. However, the reasoning presented above can be naturally extended so that any vector of class supports $[d_1(x_k),\ldots,d_M(x_k)]$ produced by the classifier ψ for x_k can be modelled. For this purpose, the parameters of the beta distributions are adjusted by solving the following system of equations for each x_k and for each $j=1,\ldots,M$

$$\mathbf{E}[\Delta_j(x_k)] = \frac{\alpha_j(x_k)}{\alpha_j(x_k)+\beta_j(x_k)} = d_j(x_k),$$

$$\alpha_j(x_k)+\beta_j(x_k) = M. \tag{11}$$

As a result, each class support $d_j(x_k)$ is modelled by the rv $\Delta_j(x_k)$ that is beta distributed with the parameters $\alpha_j(x_k)$ and $\beta_j(x_k)$ and for which conditions 1 and 2 in (6) are satisfied.

After the probability distributions are defined, the formula for the probability $P_c(RRC|x_k)$ can be obtained and subsequently, the classifier competence $C(\psi,x_k)$ for each feature vector x_k can be calculated. Assuming without loss of generality that x_k belongs to the first class, the competence $C(\psi,x_k)$ is equal to the probability that a number drawn from the beta distribution with the parameters $\alpha_1(x_k)$ and $\beta_1(x_k)$ is greater than $M-1$ numbers drawn from the beta distributions with the parameters $\alpha_j(x_k)$ and $\beta_j(x_k)$, $j=2,\ldots,M$, respectively. In order to satisfy condition 3 in (6), the M numbers drawn are normalised to sum to one, if necessary. The resulting formula for $C(\psi,x_k)$ is therefore given by

$$\begin{aligned} C(\psi,x_k) &= \int_0^1 b(u,\alpha_1(x_k),\beta_1(x_k))\left[\prod_{j=2}^{M}\int_0^u b(w,\alpha_j(x_k),\beta_j(x_k))\,dw\right]du \\ &= \int_0^1 b(u,\alpha_1(x_k),\beta_1(x_k))\left[\prod_{j=2}^{M} B(u,\alpha_j(x_k),\beta_j(x_k))\right]du, \end{aligned} \tag{12}$$

where $B(u,\alpha_j(x_k),\beta_j(x_k)) = \int_0^u b(w,\alpha_j(x_k),\beta_j(x_k))\,dw$ is a beta cumulative distribution function. Pseudocode with the steps that are required to calculate the classifier competence is given in Fig. 2. From computational perspective, it is important to notice that the beta probability distribution and the beta cumulative distribution functions are efficiently implemented in most software packages for technical computing. Using these functions, a MATLAB code `ccprmod.m` for calculating the proposed classifier competence was developed and it is freely available for download [43].

2.4. Generalising the competences

The construction of the competence function essentially depends on the interpretation of the information encapsulated in the competence set. In this study, two interpretations based on two different learning paradigms (i.e. a potential function model and regression) are used, each one of them having its own advantages and disadvantages allowing for specific practical recommendations.

2.4.1. Potential function model

In this interpretation, the feature vectors x_k are considered to be the locations of the competence sources $C(\psi_l,x_k)$ that influence the entire feature space $\mathcal{X}$, creating a competence field. The competence at x is a result of the cumulative influence of the sources, where the influence of each source is proportional to $C(\psi_l,x_k)$ and it decreases as the distance between x and x_k increases. This interpretation allows for using the potential function model [44] to construct the competence function as follows:

$$c(\psi_l,x) = \sum_{k=1}^{N} C(\psi_l,x_k)K(x,x_k), \tag{13}$$

where $K(x,x_k)$ is a non-negative potential function decreasing with the increasing distance between x and x_k. In this study, a Gaussian potential function with the Euclidean distance $K(x,x_k)=\exp(-\mathrm{dist}(x,x_k)^2)$ is used. The function was substituted into (13) which was then normalised in order for the competence function to take values in the interval [0,1]. This resulted in the

CLASSIFIER COMPETENCE

$$C(\psi, x_k)$$

1. Input the class label j_k of the feature vector x_k and the vector of class supports $[d_1(x_k),\ldots,d_M(x_k)]$ produced by the classifier ψ.
2. Calculate the parameters of the beta distributions for each $j=1,\ldots,M$

$$\alpha_j(x_k) = Md_j(x_k),$$
$$\beta_j(x_k) = M[1-d_j(x_k)].$$

3. Construct the RRC and calculate its conditional probability of correct classification

$$P_c(RRC|x_k) = \int_0^1 b(u,\alpha_{j_k}(x_k),\beta_{j_k}(x_k))\left[\prod_{\substack{j=1\\ j\neq j_k}}^{M} B(u,\alpha_j(x_k),\beta_j(x_k))\right]du.$$

4. Return the classifier competence

$$C(\psi, x_k) = P_c(RRC|x_k).$$

Fig. 2. Pseudocode of the probabilistic model of the classifier competence.

following formula:

$$c(\psi_l,x)=\frac{\sum_{k=1}^{N} C(\psi_l,x_k)\exp(-\mathrm{dist}(x,x_k)^2)}{\sum_{k=1}^{N}\exp(-\mathrm{dist}(x,x_k)^2)}. \quad (14)$$

The potential function model is easy to implement and resistant to overtraining. However, it requires the competence set to be stored and this adversely affects its use in large-scale classification problems.

2.4.2. Regression

In this interpretation, the following relation between the competences calculated for the validation set and the competence function is assumed

$$C(\psi_l,x)=c(\psi_l,x)+\varepsilon(x), \quad (15)$$

where $\varepsilon(x)$ is a random disturbance term with zero expected value $\mathbf{E}[\varepsilon(x)]=0$ for all $x\in\mathcal{X}$ and a finite variance $\sup_{x\in\mathcal{X}}\mathrm{var}(\varepsilon(x))<\infty$. This allows for defining the competence function as a regression function

$$c(\psi_l,x)=\mathbf{E}[C(\psi_l,x)|x] \quad (16)$$

and subsequently, for constructing $c(\psi_l,x)$ through an estimation of the regression function using the competence set $\mathcal{C}_l$. Assuming that the competence function depends on a finite set of parameters $\mathbf{p}=[p^{(1)},\ldots,p^{(s)}]^T$, $\mathbf{p}\in\mathcal{R}^s$ (T denotes transpose)

$$c(\psi_l,x)=c(\psi_l,x,\mathbf{p}), \quad (17)$$

the construction of the function is a parametric estimation problem which can be solved using the least squares method. The solution is an estimated competence function $\hat{c}(\psi_l,x,\mathbf{p})=c(\psi_l,x,\hat{\mathbf{p}})$, where the parameters $\hat{\mathbf{p}}$ are found by minimising the following criterion:

$$Q(\mathbf{p})=\sum_{k=1}^{N}[C(\psi_l,x_k)-c(\psi_l,x_k,\mathbf{p})]^2. \quad (18)$$

The estimated competence function is truncated if it returns values outside the interval [0,1]. For the competence function of the form

$$c(\psi_l,x,\mathbf{p})=\mathbf{p}(\psi_l)^T\varphi(x), \quad (19)$$

where $\varphi(x)=[\varphi^{(1)}(x),\ldots,\varphi^{(s)}(x)]^T$ is a vector of known transformations of x, the optimal parameter is equal to

$$\hat{\mathbf{p}}(\psi_l)=(\Phi^T\Phi)^{-1}\Phi^T C(\psi_l). \quad (20)$$

The design matrix $\Phi=[\varphi(x_1),\ldots,\varphi(x_N)]^T$ consists of the vectors $\varphi(x)$ calculated for the feature vectors x_k, $k=1,\ldots,N$ in the validation set and the vector $C(\psi_l)=[C(\psi_l,x_1),\ldots,C(\psi_l,x_N)]$ consists of the competences in the set $\mathcal{C}_l$. In this study, the above form of the competence function and three different transformation vectors $\varphi(x)$ are used. For the three vectors, each feature in x is transformed into polynomials $\sum_{i=0}^{r}x^i$ of orders $r=2,3$ and 5, respectively. The second order interaction terms between every pair of the features in x are also included in the vectors. The number of the parameters to estimate is therefore equal to $s=1+r\cdot n+n(n-1)/2$ (constant term + polynomial terms + interaction terms).

The competence function constructed using regression does not require the competence set to be stored and it is generally faster than the one constructed using the potential function model. However, if $s>N$ then the matrix $\Phi^T\Phi$ is singular and a pseudoinverse must be used to obtain the optimal parameters $\hat{\mathbf{p}}(\psi_l)$ [45]. Therefore, as the value of s increases, risk of overfitting also increases.

2.5. Examples

1. Consider a classification problem with three classes ($M=3$) where the competence $C(\psi,x_k)$ of the classifier ψ for the feature vector x_k is to be found. The class label of x_k is $j_k=2$ and the vector of class supports produced by the classifier for x_k is equal to [0.3,0.6,0.1]. First, the parameters $\alpha_j(x_k)$ and $\beta_j(x_k)$, $j=1,2,3$ are calculated using (11). The resulting values are $(\alpha_1(x_k),\beta_1(x_k))=(0.9,2.1)$, $(\alpha_2(x_k),\beta_2(x_k))=(1.8,1.2)$ and $(\alpha_3(x_k),\beta_3(x_k))=(0.3,2.7)$. The beta probability distributions with the parameters calculated are plotted in Fig. 3. Then, using (12) the competence $C(\psi,x_k)$ is obtained as

$$C(\psi,x_k)=\int_0^1 b(u,1.8,1.2)B(u,0.9,2.1)B(u,0.3,2.7)\,du\approx 0.786.$$

Therefore, the competence of the classifier ψ for the feature vector x_k is equal to 0.786.

2. This example shows the relation between the competence and a support given by the classifier ψ for the correct class. Let the classifier produce the vector of class supports $[d_1(x_k),d,\ldots,d]$, where $d=(1-d_1(x_k))/(M-1)$ and let x_k belong to the first class ($j_k=1$). The competence $C(\psi,x_k)$ plotted against $d_1(x_k)\in[0,1]$ for different values of M is shown in Fig. 4. It can be noticed from the figure that the competence is equal to the probability of random classification $C(\psi,x_k)=1/M$ if a vector of class supports is equal to $[1/M,\ldots,1/M]$.

3. Consider a classification problem with two classes ($M=2$) where feature vectors are drawn from two normal distributions with different mean vectors and covariance matrices for each class. Trained linear classifier and the validation set are

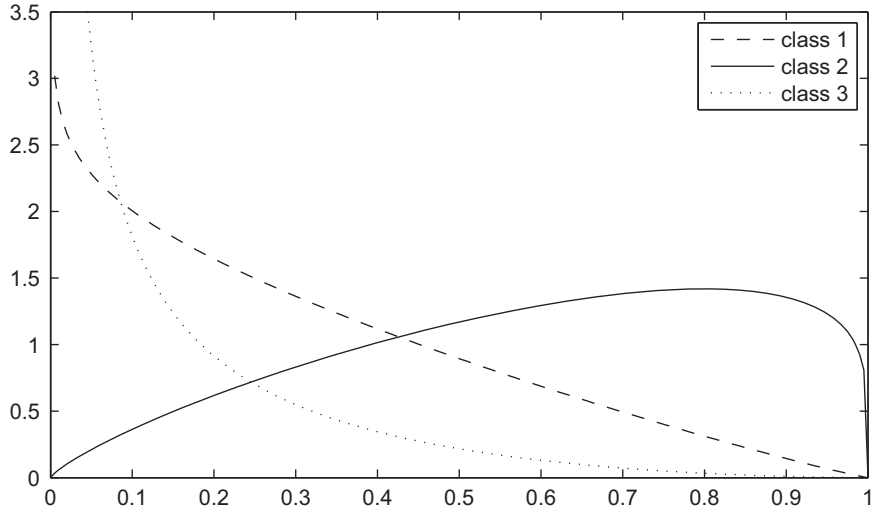

Fig. 3. The beta probability distributions of the class supports from example 1. The expected values of the distributions are equal to the class supports.

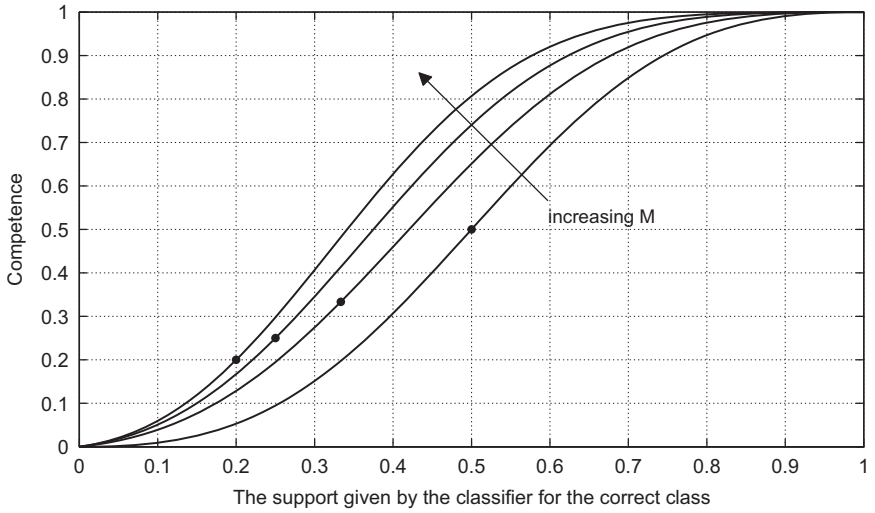

Fig. 4. The competence plotted against a support given by the classifier for the correct class (the remaining support is equally divided among incorrect classes). The class numbers are $M=2,3,4,5$ and the dots are plotted at the points $[1/M,1/M]$.

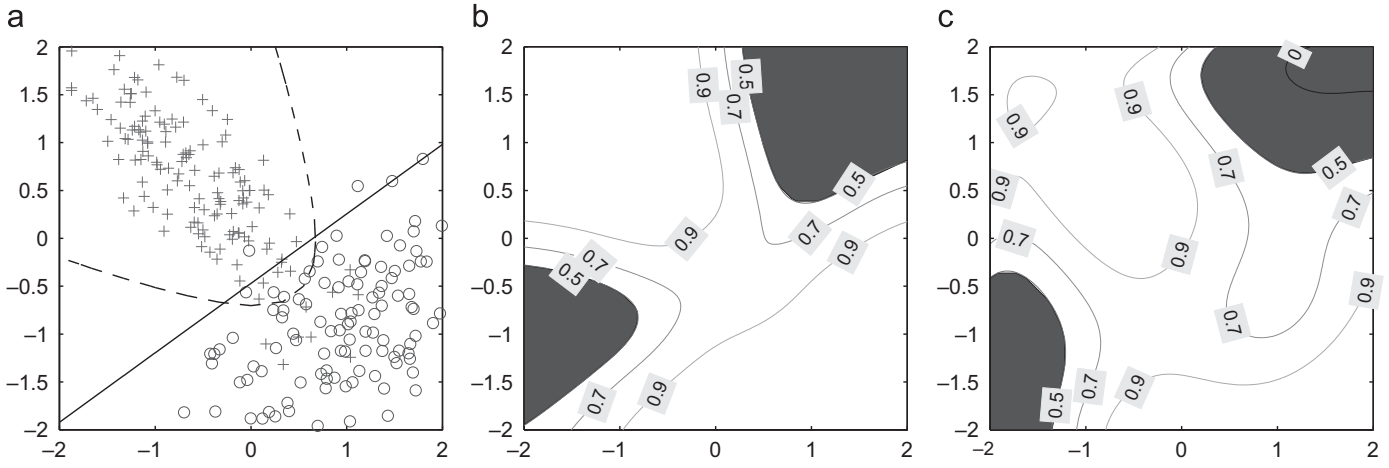

Fig. 5. (a) The validation set, the optimal decision boundary (dashed line) and the boundary obtained from the linear classifier (solid line) for the classification problem with two classes (+ and ◦) from example 3, and (b,c) two competence maps calculated for the linear classifier using the potential function model and regression, respectively; the subregions where the competences are lower than the probability of random classification (<0.5) are shaded.

given, and the goal is to calculate two competence maps for a square region in the feature space by generalising the competences calculated for the validation set to the entire feature space. For the generalisation, the potential function model with the normalised Gaussian potential function and regression with $r=5$ are used. The validation set, the optimal decision boundary and the decision boundary obtained from the linear classifier are shown in Fig. 5(a). Using (13), the competence maps were calculated and they are depicted in Figs. 5(b) and (c). The subregions where the competences are lower than the probability of random classification (<0.5) are shaded.

3. Methods

Three DCS and DES based classification systems were constructed using the probabilistic model of the classifier competence. For each of the systems, four methods of generalising the competences were evaluated, i.e. the potential function model with the normalised Gaussian potential function and regression with $r=2,3$ and 5.

1. DCS-most competent (DCS-MC): This system classifies the feature vector x in the following manner. First, the competence set $\mathcal{C}_l$ and the competence function $c(\psi_l,x)$ are constructed for each classifier in the ensemble. Then, the DCS-MC system ψ_{MC} selects the most competent classifier from the ensemble and uses it for the classification of x

$$\psi_{MC}(x,V,\Psi)=i \;\Leftrightarrow\; d_{ki}(x)=\max_{j\in\mathcal{M}} d_{kj}(x)\wedge c(\psi_k,x)=\max_{l=1,\ldots,L} c(\psi_l,x). \quad (21)$$

2. DES-competence based on label outputs and weighted majority voting (DES-CV):This system is the same as the DCS-MC system, except that a subset Ψ_x^* of the classifiers with the competences greater than the probability of random classification is selected from the ensemble for each x

$$\Psi_x^*=\{\psi_{l_1},\ldots,\psi_{l_T}\} \quad \text{where } c(\psi_{l_t},x)>\frac{1}{M}, \quad t=1,\ldots,T. \quad (22)$$

DCS-MC system

$$i = \psi_{MC}(x, V, \Psi)$$

1. Input the feature vector x, the validation set V and the classifier ensemble Ψ.
2. Generate the competence sets $\mathcal{C}_l$, $l = 1, \ldots, L$.
3. Construct the competence functions $c(\psi_l, x)$, $l = 1, \ldots, L$ using the potential function model or regression and the competence sets.
4. Select the most competent classifier ψ_k

$$c(\psi_k, x) = \max_{l=1,\ldots,L} c(\psi_l, x).$$

5. Return the class label i using the maximum rule and the vector of class supports produced by the classifier ψ_k

$$\psi_{MC}(x, V, \Psi) = i \Leftrightarrow d_{ki}(x) = \max_{j \in \mathcal{M}} d_{kj}(x).$$

Fig. 6. Pseudocode of the DCS-MC system.

DES-CV system

$$i = \psi_{CV}(x, V, \Psi)$$

1.-3. These steps are the same as in the DCS-MC system.

4. Select the subset Ψ_x^* of the classifiers with the competences greater than the probability of random classification

$$\Psi_x^* = \{\psi_{l_1}, \ldots, \psi_{l_T}\}, \text{ where } c(\psi_{l_t}, x) > \frac{1}{M}, \ t = 1, \ldots, T.$$

5. Use the weighted majority voting rule to produce a vector of class supports

$$d_j^{CV}(x) = \sum_{t=1}^{T} c(\psi_{l_t}, x) \mathbf{1}\left[d_{l_t j}(x) = \max_{k \in \mathcal{M}} d_{l_t k}(x)\right].$$

6. Return the class label i using the maximum rule and the vector of class supports produced

$$\psi_{CV}(x, V, \Psi) = i \Leftrightarrow d_i^{CV}(x) = \max_{j \in \mathcal{M}} d_j^{CV}(x).$$

Fig. 7. Pseudocode of the DES-CV system.

This step eliminates inaccurate classifiers and keeps the ensemble relatively diverse [34,35]. The selected classifiers are combined using the weighted majority voting rule [46] where the weights are equal to the competences. This results in the following vector of class supports:

$$d_j^{CV}(x) = \sum_{t=1}^{T} c(\psi_{l_t}, x) \mathbf{1}\left[d_{l_t j}(x) = \max_{k \in \mathcal{M}} d_{l_t k}(x)\right], \tag{23}$$

where $\mathbf{1}[S]$ is the indicator function equal to 1 (0) if a statement S is true (false). The DES-CV system ψ_{CV} classifies x using the maximum rule

$$\psi_{CV}(x, V, \Psi) = i \Leftrightarrow d_i^{CV}(x) = \max_{j \in \mathcal{M}} d_j^{CV}(x). \tag{24}$$

In the case where the subset Ψ_x^* is empty (i.e. there are no classifiers selected), x is assigned with a random class label.

3. DES-competence based on continuous-valued outputs and weighted class supports (DES-CS):This system is the same as the DES-CV system, except that a weighted vector of class supports is used instead of the weighted majority voting rule. The weighted vector of class supports is given by

$$d_j^{CS}(x) = \sum_{t=1}^{T} c(\psi_{l_t}, x) d_{l_t j}(x). \tag{25}$$

Again, the maximum rule is used for the classification of x

$$\psi_{CS}(x, V, \Psi) = i \Leftrightarrow d_i^{CS}(x) = \max_{j \in \mathcal{M}} d_j^{CS}(x) \tag{26}$$

DES-CS system

$$i = \psi_{CS}(x, V, \Psi)$$

1.-4. These steps are the same as in the DES-CV system.

5. Use the competence function to produce a weighted vector of class supports

$$d_j^{CS}(x) = \sum_{t=1}^{T} c(\psi_{l_t}, x) d_{l_t j}(x).$$

6. Return the class label i using the maximum rule and the vector of class supports produced

$$\psi_{CS}(x, V, \Psi) = i \Leftrightarrow d_i^{CS}(x) = \max_{j \in \mathcal{M}} d_j^{CS}(x).$$

Fig. 8. Pseudocode of the DES-CS system.

and a random class labels is assigned to x in the case where the subset Ψ_x^* is empty.

Pseudocodes of the DCS-MC, DES-CV and DES-CS systems are shown in Figs. 6, 7 and 8, respectively.

4. Experiments

Performance of the systems was evaluated in two experiments using 22 benchmark data sets. In the first experiment, the three systems constructed were evaluated using different competence generalisation methods. The system and method that showed the best combined performance were identified. In the second experiment, the system and the method identified were compared against other competence based MCSs. The experiments were conducted in MATLAB using PRTools 4.1 [47].

4.1. Data sets

The 22 benchmark data sets were taken from the UCI Machine Learning Repository [37], the Ludmila Kuncheva Collection (LKC) [38] and the ELENA project [39]. A brief description of the data sets used is given in Table 1. For each data set, feature vectors were normalised to zero mean and unit standard deviation. Two-fold cross-validation was used to extract training and testing sets from each data set. For the calculation of the competences, a two-fold stacked generalisation method was used [48]. In the method, the training set is split into two sets A and B of roughly equal sizes. The set A is first used for the training of the classifiers in the ensemble while the set B is used for the calculation of the competences. Then, the set B is used for the training while the competences are calculated using the set A. Finally, the competences calculated for both sets are stacked together and the classifiers in the ensemble are trained using the union of the sets A and B (i.e. the original training set). In this way, the competences of the classifiers are calculated for all the feature vectors in the original training set, but the data used for the calculation is unseen during the classifier training.

4.2. MCSs used for comparison

The performance of the systems constructed was compared against the following eight MCSs:

1. The single best (SB) classifier in the ensemble.
2. Majority voting (MV) of all classifiers in the ensemble.

Table 1
The data sets used in the experiments.

Data set	Source	#Objects	#Features	#Classes
Blood transfusion [49]	UCI	748	4	2
Breast cancer Wisconsin [50]	UCI	699	9	2
Clouds	UCI	5000	2	2
Dermatology	UCI	366	34	6
EColi [51]	UCI	336	7	8
Glass	UCI	214	9	6
Haberman's survival	UCI	306	3	2
Ionosphere	UCI	351	34	2
Iris	UCI	150	4	3
Laryngeal3	LKC	353	16	3
OptDigits	UCI	3823	64	10
Page blocks	UCI	5473	10	5
Parkinson [52]	UCI	195	22	2
Phoneme	ELENA	5404	5	2
Pima Indians	UCI	768	8	2
Segmentation	UCI	2310	19	7
Sonar	UCI	208	60	2
Spam	UCI	4601	57	2
Thyroid	LKC	215	5	3
Vowel	UCI	990	10	11
Wine	UCI	178	13	3
Yeast	UCI	1484	8	10

3. DCS-potential function estimate (DCS-PFE): This system classifies the feature vector x in the following manner. First, the competence $C(\psi_l,x_k)$ is assigned with the value of 1 (-1) if the classifier ψ_l correctly (incorrectly) classifies x_k taken from the validation set. Then, the competence function $c(\psi_l,x)$ is constructed using the following potential function model:

$$c(\psi_l,x) = \sum_{k=1}^{N} \frac{C(\psi_l,x_k)}{1+\mathrm{dist}(x,x_k)^2}.$$

Finally, the most competent classifier is selected from the ensemble and it is used for the classification of x [26].
4. DCS-local accuracy (DCS-LA): This system is the same as the DCS-PFE system, except the competence $C(\psi_l,x)$ is defined as a local classification accuracy. The accuracy is estimated using k nearest neighbours of x taken from the validation set. The value of $k=10$ was used for which the system achieved the best overall performance in the previous studies [27].
5. DCS-modified local accuracy (DCS-MLA): This system is the same as the DCS-LA system, except the accuracy is estimated using weighted $k=10$ nearest neighbours of x [28].

6. DCS-multiple classifier behaviour (DCS-MCB): This system is the same as the DCS-PFE system, except the competence $C(\psi_l,x)$ is defined as the classification accuracy calculated for a set $\tilde{V}$. The set $\tilde{V}$ is generated from the validation set as follows. First, a multiple classifier behaviour (MCB) is calculated for x and for its k-nearest neighbours taken the validation set. The MCB is defined as a vector whose elements are class labels assigned by all classifiers in the ensemble. Next, similarities between the MCBs are calculated using the averaged Hamming distance. Finally, the objects in the validation set with the feature vectors x_k that are most similar to x (i.e. below some similarity threshold τ) are used to generate the set $\tilde{V}$. Since the optimal values of k and τ were not given in [29], they were arbitrarily set to $k=10$ and $\tau=0.5$.
7. DES-KNORA eliminate (DES-KE): This system classifies x using the majority voting rule and a subset of classifiers selected from the ensemble. The selected subset consists of the classifiers that correctly classify k nearest neighbours of x taken from the validation set. If there are no classifiers selected, the value of k is decreased. The value of $k=8$ was used for which the DES-KE system achieved the best performance [30].
8. MoE—mixtures of experts: This system trains classifiers and calculates their competences in a coupled manner using the expectation maximisation algorithm [53]. The system is organised in a tree-like structure with the classifiers located at the terminal nodes and gating networks located at the non-terminal nodes. The gating networks are used to partition the feature space into regions of competence in order for a classification problem at hand to be simplified. The feature vector x is classified using a weighted response of the classifiers where the weights are obtained from the gating networks. The parameters of the system were set as suggested in [53,54].

4.3. Classifier ensembles

The experiments were conducted using two ensemble types: homogeneous and heterogeneous. The homogeneous ensemble consisted of 50 pruned decision tree classifiers with Gini splitting criterion. The heterogeneous ensemble consisted of the following 10 classifiers [40]: (1, 2) linear (quadratic) classifier based on normal distributions with the same (different) covariance matrix for each class, (3) nearest mean classifier, (4–6) k-nearest neighbours classifiers with $k=1,5,15$, (7, 8) Parzen density based classifier with the Gaussian kernel and the optimal smoothing parameter h_{opt} (and the smoothing parameter $h_{opt}/2$), (9) pruned decision tree classifier with Gini splitting criterion, (10) support vector machine classifier with radial basis kernel function. For both ensemble types, classifiers were trained using bootstrapping of the training set.

5. Results and discussion

Classification accuracies were averaged over 30 repetitions of two-fold cross-validation. Statistical differences in rank between the systems were obtained using a Friedman test with Iman and Davenport correction combined with a post hoc Holm's step-down procedure [55]. The average ranks of the systems and a critical rank difference calculated using a Bonferroni–Dunn test [55] were visualised. Since the two tests used may produce different results, conclusions were drawn using the Friedman test which is more powerful. The level of $p<0.05$ was considered as statistically significant.

5.1. Competence generalisation methods and combination functions

The results obtained from the first experiment for the three systems constructed and the homogeneous and heterogeneous ensembles are presented in Tables 2 and 3, respectively. The average ranks of the systems constructed and the critical rank difference (2.961) are shown in Figs. 9 and 10. For each system, the use of the potential function model resulted in the best average rank regardless of the ensemble type used. One possible reason for this is that the potential function model is not parameterised and hence it can approximate a more general class

Table 2
Classification accuracies (in per cent) and average ranks of the systems constructed for different competence generalisation methods and the homogeneous ensemble. MC, CV and CS denote the DCS-MC, DES-CV and DES-CS systems, respectively. The best result for each data set is in bold.

Data set	Gaussian potential			Regression $r=2$			Regression $r=3$			Regression $r=5$		
	MC	CV	CS	MC	CV	CS	MC	CV	CS	MC	CV	CS
Blood	76.23	76.34	76.38	76.69	76.89	76.86	76.80	77.08	**77.10**	76.55	76.73	76.79
Breast	94.19	95.67	95.67	93.98	95.66	95.67	94.05	95.63	95.67	94.11	95.60	**95.69**
Clouds	63.92	62.61	63.58	69.42	68.10	69.03	**70.39**	69.08	70.03	69.25	67.93	68.87
Dermat	57.59	80.69	**82.31**	54.95	79.14	81.07	54.87	79.20	81.12	56.02	79.33	81.25
EColi	68.31	79.89	**79.98**	67.40	79.52	79.48	67.50	79.41	79.69	67.18	79.41	79.56
Glass	54.81	66.24	**66.58**	53.37	65.06	65.50	53.73	65.07	65.68	51.51	64.85	65.60
Haber	73.32	**73.48**	73.42	72.39	72.74	72.71	72.87	73.18	73.04	73.11	73.29	73.26
Iono	84.63	86.31	**87.05**	83.98	85.74	86.40	84.33	86.24	86.81	84.26	85.91	86.60
Iris	90.96	93.00	93.04	90.64	93.07	93.29	91.42	93.20	93.31	91.29	93.29	**93.36**
Laryng	64.40	71.95	**72.24**	63.35	70.94	71.54	62.98	70.88	71.29	63.32	70.73	71.11
OptDig	51.75	84.35	86.63	51.09	84.33	**86.69**	51.15	84.17	86.57	50.99	83.90	86.40
Page	95.03	**96.18**	96.00	95.03	96.11	95.94	94.96	96.13	95.94	94.97	96.10	95.93
Parkin	83.76	87.85	**87.86**	81.21	86.36	86.23	80.87	86.45	86.48	81.76	86.62	86.63
Phoneme	75.76	**75.78**	75.77	72.97	72.98	72.98	74.03	74.04	74.05	75.37	75.38	75.42
Pima	65.60	65.56	**66.05**	65.47	65.44	65.92	65.39	65.45	66.02	65.39	65.51	65.98
Segment	83.37	**95.91**	95.64	82.95	95.75	95.58	82.82	95.72	95.55	82.75	95.71	95.54
Sonar	68.24	76.68	**76.70**	67.39	75.26	75.20	67.53	75.31	75.24	67.33	75.76	75.85
Spam	82.58	86.02	**88.89**	81.63	85.79	88.61	81.60	85.75	88.54	81.44	85.73	88.61
Thyroid	88.78	**92.67**	92.36	88.39	92.54	92.51	88.36	92.44	92.47	87.66	92.22	92.06
Vowel	53.36	78.58	**79.28**	51.41	78.69	78.96	51.64	78.47	78.88	51.55	78.51	78.75
Wine	83.13	94.85	**95.19**	81.28	94.20	94.21	81.19	93.84	93.99	82.37	94.31	94.33
Yeast	40.18	53.45	**54.25**	40.13	53.43	54.22	39.81	53.36	54.15	40.01	53.17	54.06
Avg. rank	8.59	3.95	**3.05**	10.32	6.23	4.93	10.02	6.09	4.09	10.30	6.25	4.18

Table 3
Classification accuracies (in per cent) and average ranks of the systems constructed for different competence generalisation methods and the heterogeneous ensemble. MC, CV and CS denote the DCS-MC, DES-CV and DES-CS systems, respectively. The best result for each data set is in bold.

Data set	Gaussian potential			Regression $r=2$			Regression $r=3$			Regression $r=5$		
	MC	CV	CS	MC	CV	CS	MC	CV	CS	MC	CV	CS
Blood	77.04	77.54	77.73	77.01	**78.17**	78.16	76.88	78.16	78.05	76.66	77.97	77.93
Breast	95.93	96.16	96.18	95.07	96.20	**96.30**	94.97	96.15	96.28	94.66	96.05	96.20
Clouds	**80.34**	79.71	78.88	80.03	79.69	78.94	79.99	79.65	78.80	80.06	79.80	79.00
Dermat	94.78	95.66	**95.75**	90.81	95.15	95.43	90.92	95.14	95.34	90.83	95.03	95.34
EColi	84.06	**85.74**	85.40	83.01	85.48	85.23	82.59	85.42	85.20	82.28	85.24	84.99
Glass	64.40	66.49	**66.78**	61.87	64.56	65.68	62.09	64.64	65.56	60.02	64.21	64.67
Haber	73.10	**74.25**	74.17	72.92	73.90	73.88	73.03	73.97	73.76	72.64	73.87	73.67
Iono	82.59	85.01	**86.36**	76.29	81.91	84.34	77.36	82.84	84.99	77.96	82.84	85.27
Iris	**95.80**	95.62	95.69	95.42	95.71	95.69	95.44	95.67	95.76	94.62	95.58	95.76
Laryng	70.27	**73.32**	73.21	65.27	70.11	70.62	65.74	69.29	69.42	66.21	70.43	70.34
OptDig	96.20	97.14	**97.22**	89.88	95.61	96.23	89.38	95.53	96.19	88.29	95.15	95.93
Page	96.05	95.94	96.10	95.78	95.89	**96.11**	95.82	95.88	96.10	95.81	95.90	96.10
Parkin	87.76	90.31	**90.36**	79.98	84.87	86.17	80.42	85.29	85.44	81.71	85.46	85.94
Phoneme	86.44	86.69	87.35	86.10	86.65	87.32	85.98	86.72	87.36	86.02	86.86	**87.40**
Pima	67.66	68.36	68.37	67.77	**69.03**	68.90	67.72	68.85	68.81	67.52	68.73	68.58
Segment	94.51	94.70	**95.06**	92.44	94.31	94.76	92.37	94.34	94.81	92.25	94.29	94.77
Sonar	74.81	79.52	**79.64**	67.84	73.03	73.63	66.68	71.95	73.52	67.44	72.46	74.36
Spam	90.28	91.70	**91.83**	87.89	90.41	91.09	87.89	90.42	91.03	87.41	90.10	90.73
Thyroid	92.81	92.44	92.93	91.90	92.51	**93.06**	91.58	92.50	93.01	91.09	92.06	93.01
Vowel	86.38	88.72	90.32	85.27	88.76	**90.55**	83.81	87.91	90.02	84.53	88.23	90.23
Wine	95.21	96.03	**96.35**	78.20	88.56	92.72	82.55	90.46	92.62	79.23	88.21	92.40
Yeast	56.98	**58.32**	56.82	56.52	58.30	56.75	56.29	58.21	56.79	56.37	58.27	56.82
Avg. rank	6.36	4.05	**3.27**	10.36	5.77	3.82	10.50	6.45	5.00	10.95	6.86	4.59

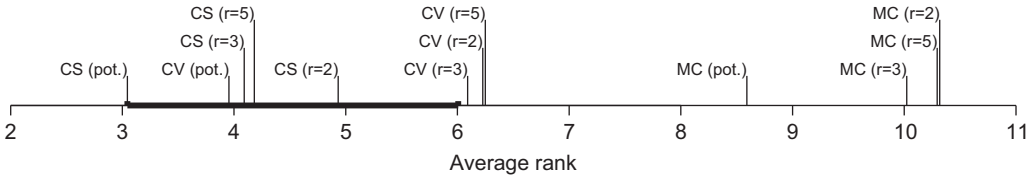

Fig. 9. Average ranks of the systems constructed for different competence generalisation methods and the homogeneous ensemble. MC, CV and CS denote the DCS-MC, DES-CV and DES-CS systems, respectively. The interval (thick line) is the critical rank difference (2.961) calculated using the Bonferroni–Dunn test ($p<0.05$).

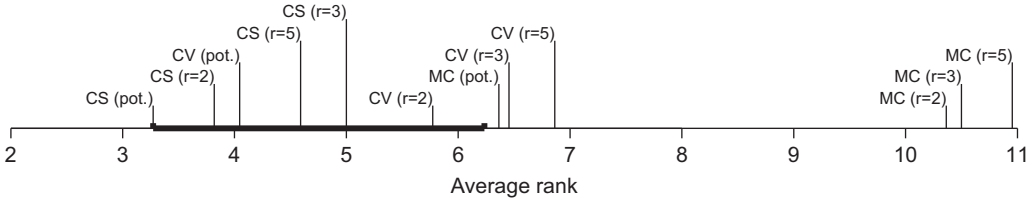

Fig. 10. Average ranks of the systems constructed for different competence generalisation methods and the heterogeneous ensemble. MC, CV and CS denote the DCS-MC, DES-CV and DES-CS systems, respectively. The interval (thick line) is the critical rank difference (2.961) calculated using the Bonferroni–Dunn test ($p<0.05$).

of functions when compared against the regression based method. Another reason could be that the model does not depend on the dimensionality of the data set and this could be to its advantage for classification problems with small sample sizes. For example, the use of the model resulted in higher classification accuracies than the use of the regression based method for the high-dimensional and small-sample-size data sets (i.e. the Dermatology, EColi, Glass, Ionosphere, Laryngeal3, Parkinson, Sonar and Wine data sets). For the regression based method, the choice of the parameter r had a relatively little effect on the results obtained, except for the heterogeneous ensemble and the DES-CV and DES-CS systems where an improvement of about one rank was obtained for the parameter $r=2$ when compared against $r=3$ and 5.

The DES-CS, DES-CV and DCS-MC systems achieved the best, the second best and the worst average ranks regardless of the competence generalisation method and the ensemble type used. This indicates that DES is superior to DCS and that neither of the competence generalisation methods used affects relative ranks of the systems constructed. Also, the combination function based on the weighted class supports used in the DES-CS system appears to be more accurate than the one based on the weighted majority voting rule used in the DES-CV system. This could be attributed to the fact that the former uses the supports for all classes, while the latter uses a single class label. Consequently, in the case where a classifier produces high supports for at least two classes, the information about "uncertainty" of the classifier is not used in the weighted majority voting rule.

From the results obtained, it can be concluded that the potential function model is the most accurate method for generalising the competences and it should be used for classification problems with small and medium sample sizes. For problems with large sample sizes, where the computational cost of the potential function model could play a role, the regression based

method with $r=2$ appears to be the most suitable one. Also, the weighted class supports and DES methodology seem to be the best choice when constructing the combination function. For these reasons, the DES-CS system based on the potential function model for generalising the competences was selected for the second experiment.

5.2. Homogeneous ensemble

The results obtained from the second experiment for the DES-CS system, the eight MCSs and the homogeneous ensemble are presented in Table 4. The system constructed had statistically significantly higher average rank than all but the MV and MoE systems. The average ranks of the systems and the critical rank difference (2.249) are shown in Fig. 11.

The SB system had the lowest average rank among all the systems evaluated, indicating that the decision tree classifiers used were unstable and diverse. This in turn could explain the better performance of the DES-CS system because the competence of an unstable classifier would be estimated more conservatively using the probabilistic model of the classifier competence when compared against the other methods. The reason is that if the support given by a classifier for an incorrect class increases, then the competence of that classifier decreases, regardless of whether the class label obtained using the maximum rule has changed. Another reason could be that the probabilistic model uses the expected value operator to ensure equivalence between the RRC and the modelled classifier. In other words, the competence is calculated using the RRC that, only on average, behaves like the modelled unstable classifier. This is likely to be a more robust approach than the one used in the other competence functions based only on the class labels returned by classifiers in the ensemble. In these functions, the information about "uncertainty" of a classifier encapsulated in its vector of class supports is not used.

The better performance of the DES-CS system could also be explained by the fact that the worse-than-random classifiers are eliminated from the ensemble before classifier combination takes place. Since inaccurate classifiers adversely affect performance of MCSs regardless of their diversity [13], thresholding the competences with the probability of random classification is a simple and effective way to select a relatively accurate and diverse subset of classifiers from the ensemble. For example, an average of 1.27% of the test feature vectors were assigned with a random class label for each data set in the experiments conducted. Therefore, all classifiers in the ensemble were evaluated as worse-than-random for these feature vectors and, assuming that the evaluation was correct, the DES-CS system outperformed all eight MCSs.

5.3. Heterogeneous ensemble

The results obtained from the second experiment for the DES-CS system, the eight MCSs and the heterogeneous ensemble are presented in Table 5. The system constructed had statistically significantly higher average rank than all other systems. The average ranks of the systems and the critical rank difference (2.249) are shown in Fig. 12.

The SB system had the best average rank among all eight MCSs. Although this indicates that classifiers in the ensemble

Table 4
Classification accuracies (in per cent) and average ranks of the DES-CS system and the eight MCSs for the homogeneous ensemble. The best result for each data set is in bold.

Data set	SB	MV	DCS-PFE	DCS-LA	DCS-MLA	DCS-MCB	DES-KE	MoE	DES-CS
Blood	76.22	76.20	76.10	76.28	76.28	76.32	76.28	**78.40**	76.44
Breast	94.27	95.77	93.66	93.76	93.83	93.96	95.29	95.41	**95.88**
Clouds	62.96	62.57	64.10	64.24	64.23	64.22	64.00	**74.42**	63.64
Dermat	58.92	84.06	57.89	58.50	77.20	58.06	62.61	**94.21**	83.91
EColi	67.56	80.06	67.99	67.00	71.51	66.92	71.43	**83.06**	81.01
Glass	50.50	65.17	52.44	52.36	55.52	52.75	57.86	65.12	**67.22**
Haber	73.45	**73.53**	72.63	72.66	72.71	72.76	73.47	72.33	73.48
Iono	84.13	86.18	84.62	84.53	84.86	84.43	85.47	83.62	**87.62**
Iris	91.70	94.10	91.07	91.10	91.07	91.13	93.07	**95.13**	94.37
Laryng	63.24	72.22	64.87	63.70	63.87	63.46	67.93	67.70	**72.69**
OptDig	51.97	**88.70**	51.69	52.01	82.10	52.18	56.21	86.74	87.35
Page	95.09	**96.09**	94.92	94.92	95.15	94.99	95.80	89.88	96.00
Parkin	82.05	87.36	82.72	82.65	82.51	82.72	85.39	80.97	**88.31**
Phoneme	70.73	70.67	70.78	70.76	70.76	70.76	70.73	**80.70**	75.95
Pima	65.62	65.20	65.55	65.68	65.61	65.31	65.49	**68.73**	66.30
Segment	82.98	**96.06**	83.60	83.41	86.98	83.52	91.32	83.98	95.72
Sonar	67.81	77.72	68.60	67.58	67.84	67.92	70.91	58.26	**78.13**
Spam	81.72	86.18	82.65	81.82	82.25	81.76	85.29	61.37	**88.85**
Thyroid	87.53	92.79	87.98	88.19	88.30	88.19	91.19	92.88	**93.21**
Vowel	49.72	79.51	50.61	50.76	74.53	52.36	55.86	**82.10**	79.45
Wine	84.02	95.68	83.65	82.15	83.17	81.78	89.52	**96.43**	95.84
Yeast	39.52	54.45	39.98	39.72	50.05	39.89	42.28	**55.85**	55.36
Avg. rank	7.14	3.27	6.48	6.68	5.16	6.27	4.23	3.82	**1.95**

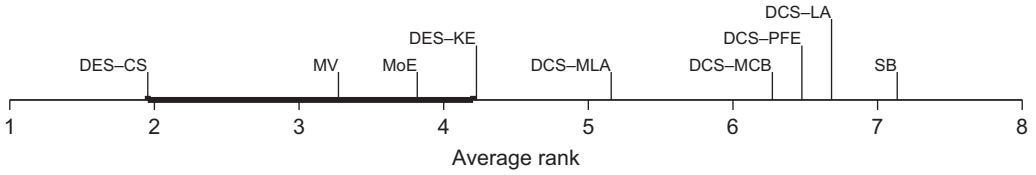

Fig. 11. Average ranks of the DES-CS system and the eight MCSs for the homogeneous ensemble. Thick interval is the critical rank difference (2.249) calculated using the Bonferroni–Dunn test ($p<0.05$).

Table 5
Classification accuracies (in per cent) and average ranks of the DES-CS system and the eight MCSs for the heterogeneous ensemble. The best result for each data set is in bold.

Data set	SB	MV	DCS-PFE	DCS-LA	DCS-MLA	DCS-MCB	DES-KE	MoE	DES-CS
Blood	77.27	77.85	77.59	76.78	76.81	75.70	76.07	**78.91**	78.26
Breast	**96.31**	96.29	96.01	96.14	96.14	95.76	95.25	95.44	96.28
Clouds	**80.44**	75.38	80.41	77.56	77.57	77.03	75.06	74.50	79.07
Dermat	95.60	96.17	95.05	95.20	95.30	95.10	95.25	93.96	**96.27**
EColi	84.78	85.72	84.63	84.09	84.37	83.19	82.43	82.97	**86.24**
Glass	66.39	64.96	64.72	64.03	62.72	64.68	64.20	65.19	**67.35**
Haber	74.29	74.00	73.41	72.66	72.73	71.52	71.54	72.76	**74.59**
Iono	84.56	84.27	85.57	84.19	83.15	83.79	83.62	83.62	**86.95**
Iris	**96.43**	95.63	95.23	95.90	95.90	95.67	94.73	94.43	95.87
Laryng	71.96	73.88	72.04	67.95	68.81	67.61	70.12	67.59	**73.90**
OptDig	96.35	97.14	96.38	96.02	96.03	95.98	96.76	86.74	**97.43**
Page	95.86	95.83	96.10	96.12	96.16	**96.27**	96.12	89.89	96.24
Parkin	89.20	89.69	88.46	88.18	87.57	88.49	88.83	81.23	**91.26**
Phoneme	86.37	85.06	86.35	85.67	85.65	86.41	86.89	80.70	**87.63**
Pima	**69.19**	68.53	67.90	67.53	67.68	67.49	67.61	68.73	68.91
Segment	93.66	94.78	93.98	94.09	94.28	94.37	94.47	84.06	**95.32**
Sonar	79.20	79.40	78.16	72.67	72.60	72.84	74.98	58.33	**80.44**
Spam	89.30	91.07	90.31	90.56	90.72	90.20	89.73	61.38	**91.91**
Thyroid	**93.95**	92.11	93.81	92.86	93.18	93.18	93.32	93.02	93.62
Vowel	86.99	87.14	88.05	84.03	82.84	84.49	84.55	82.17	**90.18**
Wine	96.60	96.69	95.09	95.62	95.87	95.56	95.56	96.82	**97.17**
Yeast	57.76	57.64	**57.93**	55.63	56.05	53.57	53.12	55.85	57.79
Avg. rank	3.45	3.82	4.59	6.11	5.70	6.32	6.23	7.18	**1.59**

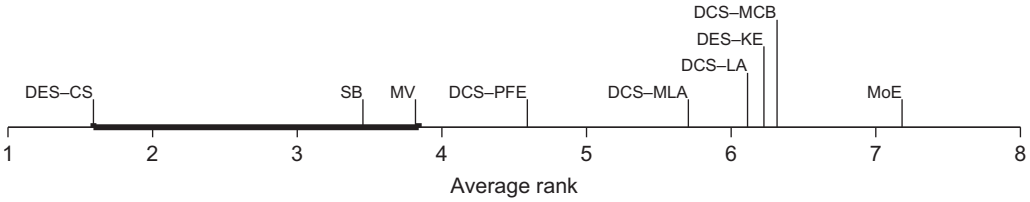

Fig. 12. Average ranks of the DES-CS system and the eight MCSs for the heterogeneous ensemble. Thick interval is the critical rank difference (2.249) calculated using the Bonferroni–Dunn test ($p < 0.05$).

were highly accurate, the ensemble was not dominated by any particular classifier type. For example, each type of classifier used, except the Parzen with the smoothing parameter $h_{opt}/2$ and the decision tree, outperformed all other classifiers in the ensemble for at least one data set. Therefore, the heterogeneous ensemble was likely to be diverse regardless of the fact that some of the classifiers were highly accurate. This observation has support in the previous study where it was shown that the CF and CS based systems perform better for heterogeneous ensembles [14]. The results obtained for the MV system, which had the second best average rank, show that the optimal subset consisted of all or nearly all classifiers in the ensemble. This could also be the reason for the better performance of the DES-CS system since it selected, on average, 8.5 out of 10 classifiers from the ensemble for each test feature vector. This in turn indicates that the probabilistic model developed can correctly evaluate the competences of highly accurate classifiers, i.e. the classifiers that produce the vectors of class supports dominated by the support given for the correct class.

For some of the data sets used, the differences in classification accuracy between the DES-CS system and the eight MCSs were relatively small. Nonetheless, even small differences obtained in favour of the system constructed for a number of data sets increase statistical significance of its average rank. This could be of practical importance in, for examples, industrial and medical applications where the costs of misclassification are high and therefore even a small improvement in performance is beneficial.

Based on the results obtained, it is suggested that the full vector of class supports should be used for evaluating the classifier competence. Since the classification errors are calculated using vectors of class supports, this suggestion may also be valid for evaluating the ensemble diversity.

6. Conclusion

In this study, a probabilistic model of the classifier competence was developed. The competence is calculated as the probability of correct classification of the randomised reference classifier that, on average, produces the same vector of class supports as the modelled classifier. Three DCS and DES based systems were constructed using the probabilistic model developed. The DES based system with the potential function model for generalising the competences outperformed eight MCSs for 22 benchmark data sets regardless of the ensemble type used (homogeneous or heterogeneous). Consequently, the model developed appears to be suitable for a wide range of classification problems.

To the best of the authors' knowledge, the proposed method of calculating the classifier competence is the first method that uses the full vector of class supports. Based on the findings of the present study, it is suggested that the development of such methods should be a focus of the future research as they potentially improve performance of multiple classifier systems.

Acknowledgement

We would like to thank the anonymous reviewers for their constructive comments and helpful suggestions.

References

[1] W. Khreich, E. Granger, A. Miri, R. Sabourin, Iterative Boolean combination of classifiers in the ROC space: an application to anomaly detection with HMMs, Pattern Recognition 43 (2010) 2732–2752.

[2] R. Polikar, J. DePasquale, H.S. Mohammed, G. Brown, L.I. Kuncheva, Learn++.MF: a random subspace approach for the missing feature problem, Pattern Recognition 43 (2010) 3817–3832.

[3] M. Ghannad-Rezaie, H. Soltanian-Zadeh, H. Ying, M. Dong, Selection-fusion approach for classification of datasets with missing values, Pattern Recognition 43 (2010) 2340–2350.

[4] T. Woloszynski, M. Kurzynski, Application of combining classifiers using dynamic weights to the protein secondary structure prediction—comparative analysis of fusion methods, in: Biological and Medical Data Analysis, Lecture Notes in Computer Science, 2006, pp. 83–91.

[5] T.G. Dietterich, Ensemble methods in machine learning, in: Proceedings of the International Workshop on Multiple Classifier Systems, Lecture Notes in Computer Science, 2000, pp. 1–15.

[6] L.I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Wiley-Interscience, 2004.

[7] Y. Freund, Y. Mansour, R.E. Schapire, Why averaging classifiers can protect against overfitting, in: Eighth International Workshop on Artificial Intelligence and Statistics, 2001.

[8] L.I. Kuncheva, A theoretical study on six classifier fusion strategies, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (2) (2002) 281–286.

[9] J. Kittler, M. Hatef, R.P.W. Duin, J. Matas, On combining classifiers, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (3) (1998) 226–239.

[10] L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) 123–140.

[11] Y. Freund, R.E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences 55 (1) (1997) 119–139.

[12] T.K. Ho, The random subspace method for constructing decision forests, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (8) (1998) 832–844.

[13] D. Ruta, B. Gabrys, Classifier selection for majority voting, Information Fusion 6 (1) (2005) 63–81.

[14] A.M.P. Canuto, M.C.C. Abreu, L.M. Oliveira, J.C. Xavier Jr., A.M. Santos, Investigating the influence of the choice of the ensemble members in accuracy and diversity of selection-based and fusion-based methods for ensembles, Pattern Recognition Letters 28 (4) (2007) 472–486.

[15] Z.-H. Zhou, J. Wu, W. Tang, Ensembling neural networks: many could be better than all, Artificial Intelligence 137 (1–2) (2002) 239–263.

[16] G. Martinez-Munoz, D. Hernandez-Lobato, A. Suarez, An analysis of ensemble pruning techniques based on ordered aggregation, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (2) (2009) 245–259.

[17] D.D. Margineantu, T.G. Dietterich, Pruning adaptive boosting, in: Fourteenth International Conference on Machine Learning, 1997, pp. 211–218.

[18] G. Martinez-Munoz, A. Suarez, Aggregation ordering in bagging, in: IASTED International Conference on Artificial Intelligence and Applications, 2004, pp. 258–263.

[19] R.E. Banfield, L.O. Hall, K.W. Bowyer, W.P. Kegelmeyer, Ensemble diversity measures and their application to thinning, Information Fusion 6 (1) (2005) 49–62.

[20] G. Rogova, Combining the results of several neural network classifiers, Neural Networks 7 (5) (1994) 777–781.

[21] J. Meynet, J.-P. Thiran, Information theoretic combination of pattern classifiers, Pattern Recognition 43 (2010) 3412–3421.

[22] I. Partalas, G. Tsoumakas, I. Vlahavas, Pruning an ensemble of classifiers via reinforcement learning, Neurocomputing 72 (7–9) (2009) 1900–1909.

[23] Y. Zhang, S. Burer, W.N. Street, Ensemble pruning via semi-definite programming, Journal of Machine Learning Research 7 (2006) 1315–1338.

[24] L.I. Kuncheva, Clustering-and-selection model for classifier combination, in: Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies, 2000, pp. 185–188.

[25] R. Liu, B. Yuan, Multiple classifiers combination by clustering and selection, Information Fusion 2 (3) (2001) 163–168.

[26] L.A. Rastrigin, R.H. Erenstein, Method of collective recognition, Energoizdat, Moscow, 1981 (in Russian).

[27] K. Woods, W.P. Kegelmeyer Jr., K. Bowyer, Combination of multiple classifiers using local accuracy estimates, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (4) (1997) 405–410.

[28] P.C. Smits, Multiple classifier systems for supervised remote sensing image classification based on dynamics classifier selection, IEEE Transactions on Geoscience and Remote Sensing 40 (4) (2002) 801–813.

[29] G. Giacinto, F. Roli, Dynamic classifier selection based on multiple classifier behaviour, Pattern Recognition 34 (2001) 1879–1881.

[30] A.H.R. Ko, R. Sabourin, A.S. Britto Jr., From dynamic classifier selection to dynamic ensemble selection, Pattern Recognition 41 (5) (2008) 1718–1731.

[31] E.M.D. Santos, R. Sabourin, P. Maupin, A dynamic overproduce-and-choose strategy for the selection of classifier ensembles, Pattern Recognition 41 (10) (2008) 2993–3009.

[32] R.A. Jacobs, M.I. Jordan, S.J. Nowlan, G.E. Hinton, Adaptive mixtures of local experts, Neural Computation 3 (1991) 79–87.

[33] G. Giacinto, F. Roli, A theoretical framework for dynamic classifier selection, in: Fifteenth International Conference on Pattern Recognition, 2000, pp. 8–11.

[34] T. Woloszynski, M. Kurzynski, On a new measure of classifier competence applied to the design of multiclassifier systems, in: Image Analysis and Processing—ICIAP 2009, Lecture Notes in Computer Science, 2009, pp. 995–1004.

[35] T. Woloszynski, M. Kurzynski, On a new measure of classifier competence in the feature space, in: Computer Recognition Systems 3, Advances in Soft Computing, 2009, pp. 285–292.

[36] T. Woloszynski, M. Kurzynski, A measure of competence based on randomized reference classifier for dynamic ensemble selection, in: Twentieth International Conference on Pattern Recognition, 2010, pp. 4194–4197.

[37] A. Asuncion, D.J. Newman, UCI machine learning repository URL 〈http://www.ics.uci.edu/~mlearn/MLRepository.html〉, 2007.

[38] L.I. Kuncheva, Ludmila kuncheva collection URL 〈http://www.bangor.ac.uk/~mas00a/activities/patrec1.html〉, 2004.

[39] E.E. project: ROARS, Phoneme database from elena project, URL 〈http://www.dice.ucl.ac.be/neuralnets/Research/Projects/ELENA/〉.

[40] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, Wiley-Interscience, 2001.

[41] J.O. Berger, Statistical Decision Theory and Bayesian Analysis, Springer Verlag, New York, 1985.

[42] H.A. David, H.N. Nagaraja, Order Statistics, Wiley-Interscience, 2003.

[43] T. Woloszynski, Classifier competence based on probabilistic modeling (ccprmod.m) at Matlab central file exchange URL 〈http://www.mathworks.com/matlabcentral/fileexchange/28391-classifier-competence-based-on-probabilistic-modeling〉, 2010.

[44] W. Meisel, Potential functions in mathematical pattern recognition, IEEE Transactions on Computers C-18 (1969) 911–918.

[45] C.L. Lawson, R.J. Hanson, Solving Least Squares Problems, Society for Industrial Mathematics, 1987.

[46] E. Hullermeier, S. Vanderlooy, Combining predictions in pairwise classification: an optimal adaptive voting strategy and its relation to weighted voting, Pattern Recognition 43 (2010) 128–142.

[47] R.P. Duin, P. Juszczak, P. Paclik, E. Pekalska, D. de Ridder, D. Tax, S. Verzakov, PR-Tools4.1, a matlab toolbox for pattern recognition, 2007.

[48] D.H. Wolpert, Stacked generalization, Neural Networks 5 (2) (1992) 241–260.

[49] I.-C. Yeh, K.-J. Yang, T.-M. Ting, Knowledge discovery on RFM model using Bernoulli sequence, Expert Systems with Applications 36 (3) (2009) 5866–5871.

[50] W.H. Wolberg, O.L. Mangasarian, Multisurface method of pattern separation for medical diagnosis applied to breast cytology, Proceedings of the National Academy of Sciences, U.S.A. 87 (1990) 9193–9196.

[51] R. King, A. Karwath, A. Clare, L. Dehaspe, The utility of different representations of protein sequence for predicting functional class, Bioinformatics 17 (5) (2001) 445–454.

[52] M.A. Little, P.E. McSharry, D.A.E. Costello, I. Moroz, Exploiting nonlinear recurrence and fractal scaling properties for voice disorder detection, Biomedical Engineering Online 6 (23) (2007).

[53] M.I. Jordan, R.A. Jacobs, Hierarchical mixtures of experts and the EM algorithm, Neural Computation 6 (2) (1994) 181–214.

[54] S.R. Waterhouse, A.J. Robinson, Classification using hierarchical mixtures of experts, in: Neural Networks for Signal Processing [1994] IV, Proceedings of the 1994 IEEE Workshop, 1994, pp. 177–186.

[55] J. Demsar, Statistical comparison of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.

Tomasz Woloszynski received M.Sc. degree in computer science from the Wroclaw University of Technology (Poland) in 2005. He is currently a double Ph.D. candidate in computer science at the Wroclaw University of Technology, Poland and in engineering at the University of Western Australia, Australia. His research is focused mainly on pattern recognition, information fusion and numerical methods for medical applications.

Marek Kurzynski received Ph.D. in automatic control from the Wroclaw University of Technology (Poland) in 1974 and D.Sc. degree in computer science from the Silesian University (Poland) in 1987. He is currently a Full Professor in the Department of Systems and Computer Networks, Wroclaw University of Technology, Poland. Prof. Kurzynski has been the recipient, investigator and co-investigator of numerous research grants from EU, Polish Ministry of Science and other industrially founded research projects. He has published more than 180 refereed journal and conference papers and four books. He is an Associated Editor of the Journal of Applied Computer Science, the Int. Journal of Applied Mathematics and Computer Science and the Int. Journal of Machine Graphics and Vision. He was also one of the editors of the book series on Computer Recognition Systems (2005, 2007, 2009, 2011) published by Springer Verlag. His research interests include pattern recognition, artificial intelligence methods, biological signal processing and medical informatics.